Implementation of Control Techniques to make dynamic on-street Parking Pricing in order to reduce waiting time and traffic jam in high density areas.

Exam - CMC-12
Prof. Dr. Marcos Máximo
Instituto Tecnológico de Aeronáutica - ITA

July 17, 2024

Group Components
Gabriel S. Tedesco: gabriel.tedesco.8723@ga.ita.br
Victor A. B. Caus: victor.caus.8815@ga.ita.br
Gabriel C. Bomfim: gabriel.bomfim.8860@ga.ita.br

Abstract

This study investigates the use of a PI controller with anti-windup for dynamically adjusting parking prices to achieve a target occupancy rate of 70%. Utilizing advanced optimization techniques, including CMA-ES and Nelder-Mead, the controller was tuned to balance stability and performance metrics. Simulation results demonstrate its robustness against varying demand noise, effectively maintaining desired occupancy levels and adjusting prices over time. While not achieving initial phase and gain margin targets, the bandwidth requisite was satisfied and the the controller's implemented margins proved sufficient for stable operation, showing promise for real-world application in optimizing urban parking management systems.

***Keywords* – Dynamic parking pricing, PI controller, Anti-windup, Optimization techniques, Urban parking management**

I. THEORETICAL INTRODUCTION AND MOTIVATION

A. Motivation: The low parking cost problem

"How Much Should a Parking Space Cost?" [1] is a 10 minutes video by the YouTube channel City Beautiful that delves into the economics of parking. In the video, Dave Amos explores the implications of free and paid on-street parking, highlighting that correctly pricing parking can reduce traffic, lower greenhouse gas emissions, and economically benefit nearby businesses. These insights are supported by various studies [2]. The video also concludes that the ideal pricing strategy is dynamic, as exemplified by the SFpark program, managed by the San Francisco Municipal Transportation Agency (SFMTA), which adjusts prices based on demand to effectively address parking issues.

> The SFMTA uses demand-responsive pricing to open up parking spaces on each block and reduce circling and double-parking. Using meter payment data to estimate parking occupancy, the SFMTA raises the rate by $0.25 on blocks where average occupancy is above 80%, lowers the rate $0.25 on blocks where average occupancy is below 60%, and does not change the rate on blocks that hit the target occupancy between 60% and 80%. Rates may vary by block, by time of day, and weekday or weekend. [3]

The proposed solution by SFMTA works, but can be improved in several ways. Therefore, This study aims to develop a control system that dynamically adjusts parking prices to achieve target occupancy rates, similar to the approach used by SFpark, improving the approach for that problem. However, unlike the San Francisco project, our method will employ control system techniques to accomplish this.

The techniques employed are explained in the next sections. For that control system, it was necessary to use a PI controller under the requisites such as phase margin, gain margin and bandwidth. As the controller had a integrator, it was necessary to put an anti-windup to prevent undesired behavior of the system. Also, to best fit to the requisistes two methods of optimization were used, the Nelder-mead and the CMA-ES [10], both already implemented in matlab, with the appropriate licenses applied.

B. PI Controller

A Proportional-Integral (PI) controller is a type of feedback controller widely used in industrial control systems. It combines two terms: one proportional to the error signal and another to the integral of the error signal. The PI controller's primary advantages are:

- **Improved Steady-State Accuracy**: By integrating the error, the PI controller can eliminate the steady-state error.
- **Robustness Against Noise**: The integral action helps in filtering out noise and small disturbances.

C. Phase Margin

Phase margin is a measure of the stability of a control system. It is defined as the amount of additional phase lag required to bring the system to the verge of instability. Mathematically, it is the difference between the phase angle at the gain crossover frequency and -180 degrees. A higher phase margin indicates a more stable system and greater tolerance to delays and changes in system dynamics.

D. Gain Margin

Gain margin indicates how much the system gain can increase before the system becomes unstable. It is the factor by which the gain can be multiplied before the loop gain reaches unity at the phase crossover frequency. A higher gain margin signifies a more robust system capable of withstanding larger variations in gain without becoming unstable.

E. Bandwidth

Bandwidth refers to the range of frequencies over which the system can respond effectively to an input signal. It is typically defined as the frequency range within which the system's output power is greater than half of its maximum power. Higher bandwidth implies faster system response and better performance in tracking rapid changes in the input signal.

F. Windup Effect and Anti-Windup Heuristic

Windup occurs in PI controllers when the actuator saturates, causing the integral term to accumulate excessively. This can lead to significant overshoot and instability. The anti-windup heuristic mitigates this by limiting the integral term, typically through the introduction of a saturation block in the compensator. By doing so, it prevents the integral term from accumulating beyond the actuator limits. This is particularly important in PI controllers, as the integral component continues to integrate the error even when the actuator output is saturated, leading to an excessive buildup.

G. Nelder-Mead Optimization

The Nelder-Mead optimization is a simplex-based method for finding the minimum or maximum of an objective function in a multidimensional space. It does not require the calculation of gradients, making it suitable for non-differentiable or noisy objective functions. The method iteratively refines a simplex of points to approach the optimum. It is particularly useful in low-dimensional optimization problems where the objective function is complex and may contain many local minima.

H. CMA-ES Optimization

Covariance Matrix Adaptation Evolution Strategy (CMA-ES) is an evolutionary algorithm for difficult non-linear, non-convex optimization problems. It adapts the covariance matrix of a distribution of solutions, enabling efficient exploration of the search space. CMA-ES is particularly effective in high-dimensional spaces and has strong global optimization properties. It excels in situations where the landscape of the objective function is rugged or deceptive, providing robust convergence to global optima.

II. DYNAMIC SYSTEM MODELING

A. Plant Modeling

The schedule control system will employ a unified framework to manage various time intervals. This method allows the parking system to adapt to typical traffic patterns throughout the day, thereby enhancing the city's overall organization concerning congestion management. Each specific time slot and clearly defined parking zone, referred to in this report as a "Block," will be adjusted accordingly.

As observed in the reference study, numerous factors influence parking demand, and it is impractical to model all of them comprehensively. Therefore, for simplicity and to fit a control model, we will consider a linear interpolation based on experimentally obtained data. The controller design utilizes the following plant:

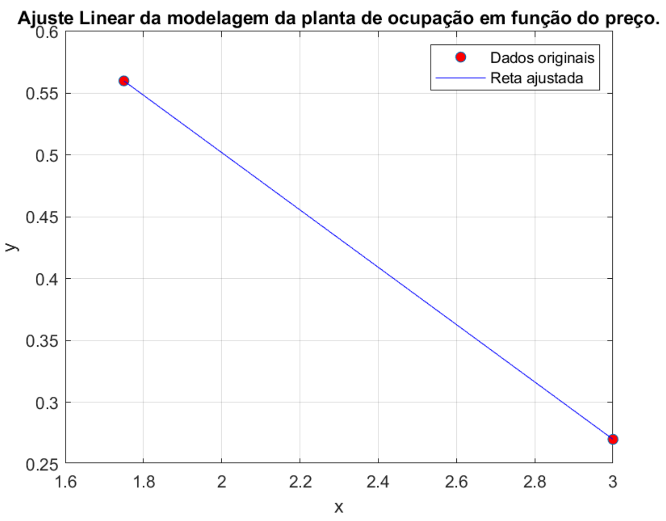

Figure 1 – Graphical representation of the model of the plant, using data of 600 block of Beach Street, Fisherman's Wharf.

It is considered there is no delay between price changes and occupancy adjustments, because we assume that the perception time of individuals is faster than the zero-hold period adopted, which is one day. This assumption is based on the premise that people generally respond to parking price changes within the same day due to the immediate and noticeable impact on their commuting and parking habits. Thus, the system can be considered effectively real-time for practical purposes.

B. Requirements Definition

First, it's important to understand that our elementary plant, which intrinsically attempts to model demand based on price, can never represent all the numerous other variables that significantly impact demand more than price itself. As "Getting the Prices Right" [2] highlights, "The wide range of price elasticities suggests, as one would expect, that many variables other than price affect demand." Therefore, our controller needs to be robust and not overly dependent on the plant used in the project. Consequently, we require a minimum phase margin and gain margin. To achieve this, we will use benchmarks based on control systems in industry for other applications, as we lack examples of control systems applied to this specific type of pricing.

Moreover, we aim to avoid disrupting people's lives; thus, we want changes to occur gradually, allowing everyone to adapt. This gradual change should encourage people to avoid driving rather than flooding the streets upon realizing prices have increased. Therefore, we desire a slow bandwidth.

To ensure consistency, we will average the daily data for each selected time interval. Each time interval will have its own separate controller. Therefore, the sampling frequency is set to a period of one day. By averaging the daily data, we can smooth out fluctuations and make more informed control decisions, leading to a more stable and predictable system. This approach helps maintain consistency and reliability in the control process, aligning with our goal of minimizing disruptions to people's daily routines.

Let PM be Phase Margin, GM be the Gain Margin, ω_b be the bandwidth,f_s be the sampling frequency, the requirements of the project are:

- $PM >= 120$
- $GM >= 30dB$
- $\omega_b <= 2.4241 \cdot 10^{-6} Hz$
- $f_s = 7.2722 \cdot 10^{-5} Hz$, that is, $T_s = 86400s$ (1 day)

C. Choice of the Controller

The selection of a Proportional-Integral (PI) controller is justified by the complexity and inherent imprecision in modeling the plant for this system. The plant parameters 'a' and 'b' in our linearization can be considered highly noisy due to the numerous variables affecting demand that cannot be precisely accounted for. This noise can be categorized into two types: large-scale noise, representing significant changes in demand trends that drive consistent price adjustments, and small-scale, high-frequency noise, which we aim to filter out.

The integrator term in the PI controller serves as an effective noise filter, particularly for high-frequency disturbances. Additionally, it helps in reducing the steady-state error, ensuring that persistent deviations are corrected over time. The integral action values the cumulative effect of past days rather than just the immediate previous day or instantaneous variations. This aligns with our conservative approach, aiming for gradual changes to help people adapt to new prices without sudden disruptions.

We prioritize a conservative and steady adaptation process over rapid responsiveness, which is why the integrator term is particularly suitable. This ensures that price adjustments are smooth and predictable, allowing individuals ample time to adapt to new parking prices without causing sudden shocks to their routines.

III. CONTROLLER PROJECT

A. Analytical controller

Before the in fact developing of the price controller, it was needed a first guess to the system. So, in the first stage of the project, it was needed to develop a analytical controller that was good with all the requisites listed before, because from that point, the other computational methods will find the optimal solutions for the problem.

Therefore, in the first place, the project of the plant is needed. The Figure 2 shows a simple diagram of the plant. The entry is the reference for the percentage of parking lots occupied that is optimum for the city, a number between 60% and 80% so that everyone knows that there will be space for car parking or the citizen just take the option, when all lots are occupied, for not to go out home by car, because there will be no parking space, preventing the overall traffic jam becoming greater.

The controller will determine the price, which is controlled by a PI, with a proportional and a integrator.

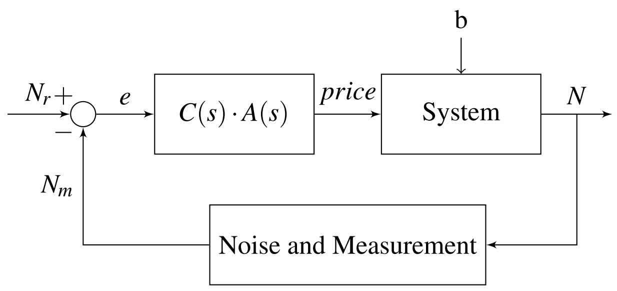

Figure 2 – Closed loop control system that was developed for this project. C(s) stands for the controller, A(s) stands for the delay, the system is a simple linear product, and b is a perturbation introduced. The noise and measurement are the feedback of N. The noise stands for the randomized aspect of the problem, and the measurement means that we are taking the mean of all cars parked, such that N_m is the mean N measured.

Looking at the system, the analytical controller was designed using the initial requirements of phase margin and bandwidth. To do this, two equations were found, in which K_i and K_p were obtained in order to meet the controller's initial prerequisites by intersecting the two requirement curves, in terms of phase margin and bandwidth. After this, the gain margin was only taken into account in the computational optimization process, due to the following factors:

Phase Margin Calculation

Calculation of ω_{cp}:

$$\left| \left(K_p + \frac{K_i}{s} \right) \cdot a \right| = 1$$

$$\left| K_p^2 \cdot \omega_{cp}^2 + K_i^2 \right| \left| a^2 \right| = \left| \omega_{cp}^2 \right|$$

$$\omega_{cp} = \sqrt{\frac{K_i^2 \cdot a^2}{1 - K_p^2 \cdot a^2}}$$

Calculation of the phase margin:

$$\angle G_a(\omega_{cp} \cdot j) + 180 = PM$$

$$PM - 180 = \angle G(\omega_{cp} \cdot j)$$

$$PM - 180 = -90 + \tan^{-1}\left(\sqrt{\frac{K_p^2 \cdot a^2}{1 - K_p^2 \cdot a^2}} \right) - K_i \cdot |a| \sqrt{\frac{1}{1 - K_p^2 \cdot a^2}}$$

$$K_i = \left(90 - PM + \tan^{-1}\left(\sqrt{\frac{K_p^2 \cdot a^2}{1 - K_p^2 \cdot a^2}} \right) \right) \sqrt{\frac{1 - K_p^2 \cdot a^2}{a^2}} \qquad (I)$$

For $\omega_b = 2.4241 \times 10^{-6}$ Hz, we have the following constraint:

$$\left| G_f(\omega_b \cdot j) \right| = \sqrt{2} \cdot \left| G_f(0) \right| = \sqrt{2}$$

$$G_f(\omega_b \cdot j) = \frac{C(s)\cdot A(s)\cdot a}{1 + C(s)\cdot A(s)\cdot a}$$

$$\left|\frac{C(s)\cdot A(s)\cdot a}{1 + C(s)\cdot A(s)\cdot a}\right| = \sqrt{2}$$

$$\left|1 + \frac{1}{C(s)\cdot A(s)\cdot a}\right| = \frac{1}{\sqrt{2}}$$

In order to provide a good initial condition for the optimization carried out using the methods known as Nelder-Mead and CMA-ES, it was considered that, by the fact that the bandwidth expected is to low and in order to simplify the math, the above expression is approximately equal to:

$$\left|\frac{1}{C(s)\cdot A(s)\cdot a}\right| = \frac{1}{\sqrt{2}}$$

Obtaining:

$$(K_p \cdot \omega_b)^2 + K_i^2 = 2 \cdot \frac{\omega_b^2}{a^2}$$

Isolating K_i as a function of K_p:

$$K_i = \sqrt{2\left(\frac{\omega_b}{a}\right)^2 - (K_p \cdot \omega_b)^2} \qquad (II)$$

So, the functions (I) and (II) of Ki in terms of Kp was plotted and the intersection, founded.

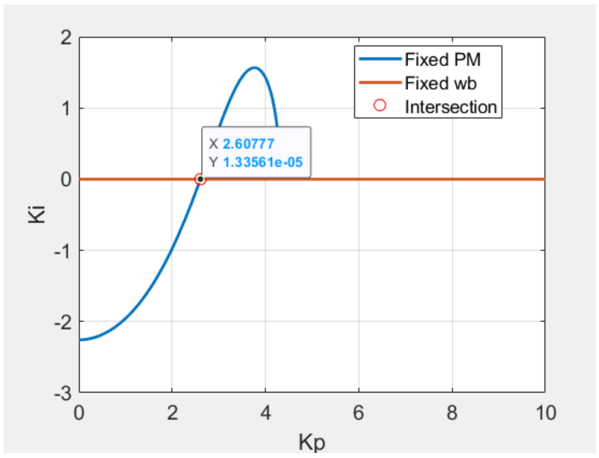

Figure 3 – Plot of the curves with fixed PM and fixed wb(aproximated) such that the first terms for Ki = 1.33561e-05 and Kp = 2.60777 are founded.

B. Controller Optimization with Nelder-Mead and CMAES

After designing the controller analytically, the analytical result was used as an initial guess for both optimizers: Nelder-Mead and CMAES. The convergence of the algorithms was tracked:

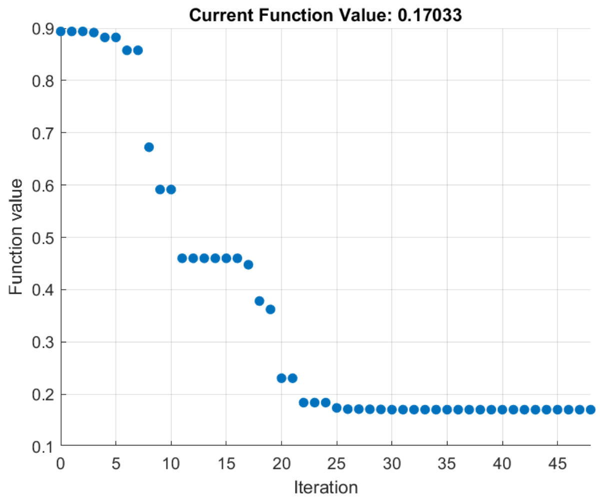

Figure 4 – Convergence of the Nelder-Mead algorithm. Both in terms of the number of iterations (it took 25 to converge) and execution time, it proved to be faster. Note that it was not possible to obtain a cost function value of 0, meaning the requirements were not completely met simultaneously.

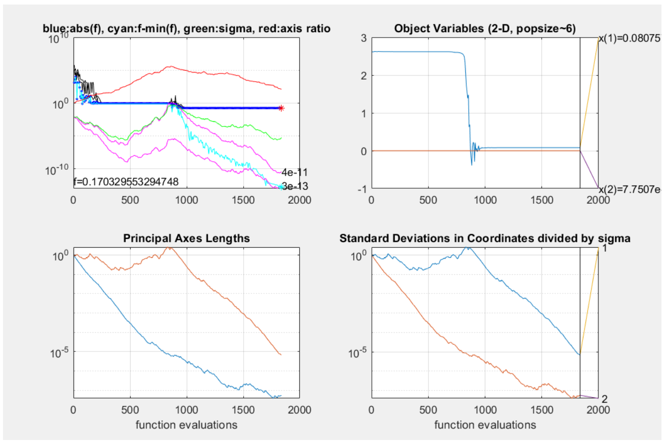

Figure 5 – Convergence of the CMAES algorithm. It took longer to converge, with 1000 iterations. In these graphs, focus on the blue line in the top-left graph, which represents the absolute value of the cost function decreasing as it converges, as well as the top-right graph, which represents the values of K_p and K_i adjusting throughout the algorithm. The values after the end (post the black vertical line) are meaningless. Note: K_i is not zero, it is just of a very small magnitude. This is expected, as we are dealing with periods of weeks rather than seconds.

The convergence behavior of the Nelder-Mead and CMAES algorithms provides valuable insights into the optimization process and the nature of our control problem.

The Nelder-Mead algorithm converged significantly faster than the CMAES algorithm, taking only 25 iterations compared to CMAES's 1000 iterations. This rapid convergence can be attributed to the simplicity and efficiency of the Nelder-Mead method for low-dimensional optimization problems. Nelder-Mead is a direct search method that does not require gradient information, making it well-suited for problems where the cost function is smooth and the dimensionality is low.

On the other hand, CMAES is a more complex and general optimization algorithm. It uses a population-based approach and adapts the covariance matrix to learn the shape of the cost function landscape, which makes it particularly effective for high-dimensional and rugged optimization problems. The longer convergence time of CMAES in this case does not imply it is a poor algorithm; rather, it reflects its robustness and adaptability to a wide range of optimization scenarios. CMAES is considered state-of-the-art for many complex optimization problems, especially those with higher dimensions or more complicated landscapes.

Despite their different approaches, both algorithms converged to the same values of K_p and K_i, which underscores the quality of our analytical design. The fact that both methods, starting from the same initial guess, arrived at the same solution suggests that the analytical approach was well-founded and that the identified minima are likely the best achievable under the given cost function. This convergence indicates that our analytical design is robust and there are no significant alternative minima that could better satisfy the requirements.

However, it is important to note that neither algorithm was able to achieve a cost function value of zero, with The cost function is defined for both methods as:

$$\begin{aligned}\text{cost} = &\left(\min\left(\frac{\text{Pm} - \text{req.PM}}{\text{req.PM}}, 0\right)\right)^2 \\ &+ \left(\frac{\omega_b - \text{req.}\omega_b}{\text{req.}\omega_b}\right)^2 \\ &+ \left(\min\left(\frac{\text{Gm} - \text{req.GM}}{\text{req.GM}}, 0\right)\right)^2\end{aligned}$$

where:

$$\begin{aligned}\text{Pm} &: \text{Phase margin} \\ \text{Gm} &: \text{Gain margin} \\ \omega_b &: \text{Bandwidth} \\ \text{req.PM} &: \text{Required phase margin} \\ \text{req.GM} &: \text{Required gain margin} \\ \text{req.}\omega_b &: \text{Required bandwidth}\end{aligned}$$

The minimum value for the cost function obtained was 0.17. This outcome highlights the stringent nature of our requirements; the simultaneous satisfaction of all criteria might be inherently infeasible given the constraints and the nature of the problem. Nonetheless, the solutions obtained are satisfactory and indicate that our design is quite close to meeting the desired performance.

C. Anti-windup Heuristic

Windup occurs in PI controllers when the system is discrete, and the integral term accumulates excessively due to the delays introduced by the discretization process. This can lead to significant overshoot and instability. The anti-windup heuristic mitigates this by limiting the integral term, typically through the introduction of a saturation mechanism that prevents the integral term from accumulating beyond certain limits.

In our case, the use of a PI controller is prone to windup primarily because of the discretization with an update rate of once per day. This significantly increases the delay of the zero-order hold, exacerbating the windup effect. Instead of using a separate saturation block, we implement the anti-windup mechanism directly in the control algorithm code. The following MATLAB function demonstrates this implementation:

```matlab
function u = compensador(e, ea, ua, Kp, Ki, T)
% u = compensador(e, ea, ua, Kp, Ki, T)
% Implements the discrete PI compensator.
% Inputs:
% e: current error e[k].
% ea: previous error e[k-1].
% ua: previous command u[k-1].
% Kp: proportional gain.
% Ki: integral gain.
% T: sampling time.
% Outputs:
% u: current command u[k].

u = ua + (Kp + Ki*T/2)*e + (-Kp + Ki*T/2)*ea;

% Saturation mechanism to prevent windup:
u = min(u, 0); % prevent a negative price (as it
    does not make sense in the context)
end
```

Listing 1: Anti-windup Heuristic implemented on the Compensator block in the Simulink archive.

In this implementation, the control signal u is computed based on the current error e, the previous error ea, and the previous control command ua. The proportional (K_p) and integral (K_i) gains, along with the sampling time T, determine the contribution of the current and previous errors to the control signal.

To prevent windup, a saturation mechanism is directly implemented in the code. This mechanism ensures that the control signal u does not go below zero, as a negative price is not meaningful in this context. By limiting the control signal in this way, the integral term is prevented from accumulating excessively, thereby maintaining the stability and performance of the PI controller even with the daily update rate and associated delay.

IV. RESULTS AND DISCUSSION

The optimized controller was evaluated using two different approaches. In the first approach, the discrete controller is approximated to a continuous controller with a delay equivalent to half the sampling period, and the plant is simplified to a multiplication by -a. A unit step input is then simulated in the simplified system, and its Bode plot, phase margin, gain margin, and bandwidth are analyzed. In the second approach, the optimized controller is simulated with greater precision, taking into account daily random movement trends represented as Gaussian noise and employing the complete PI controller with anti-windup.

A. Analytical

First, the analytical controller, projected by the Kp and Ki founded with the curves of PM = 120 and $\omega_b = 2.4241 \cdot 10^{-6} Hz$, was tested, to see if the approximation did in the analytical project corresponded to the requisites. The following graph was founded:

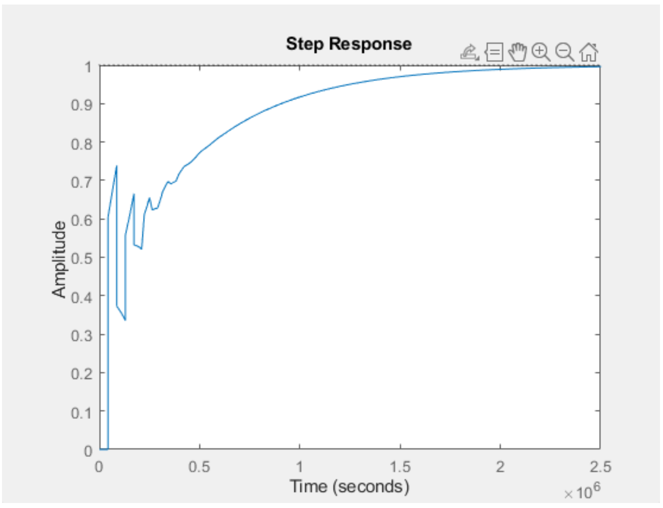

Figure 6 – Unit step response for the analytical system.

Note that the step response for the projected Kp and Ki provides an initially unstable system. But, that after some time, the convergence occurs. Analysing the phase and gain margins for that system, we have:

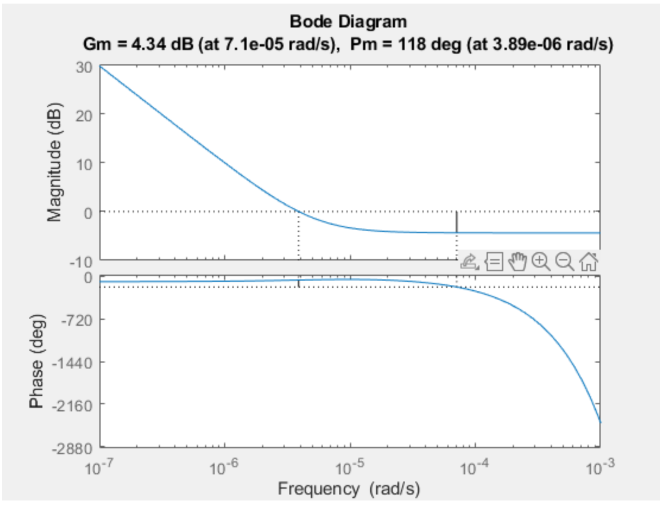

Figure 7 – Gain and phase margins for the analytical system.

Note that the approximation for the calculus of phase margin and bandwidth was effective, because the phase margin founded was 118 degrees, almost 120, and the bandwidth founded was 2.4655e-06 Hz, much close to the reference $\omega_b = 2.4241 \cdot 10^{-6} Hz$. Therefore, the analytical controller stands a nice initial point for the optimization process, which was done after this point. Once the controller PI was projected and defined to be Kp: 0.0807 and Ki: 7.7505e-06, the tests was done to that optimal system:

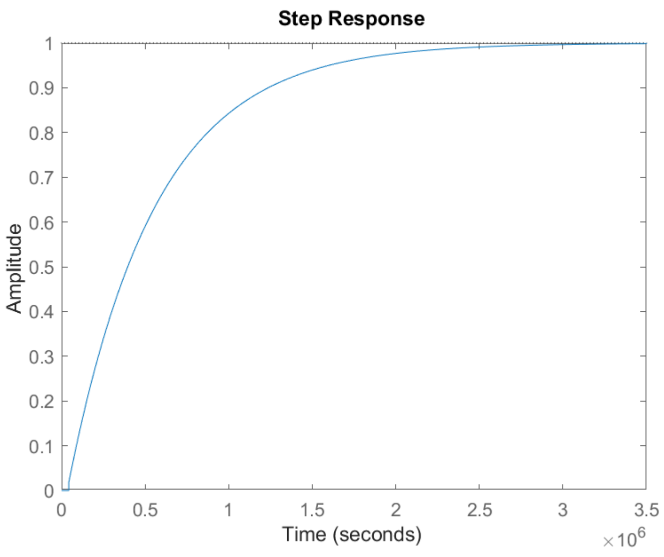

Figure 8 – Unit step response

Note that the step response of the optimal system founded by the convergence from both CMA-ES and Nelder-Mead did was smooth and achieve what was expected, a slower and gradual change of prices through time, achieving the step reference after a time good for the scale of the problem, $3 \cdot 10^6 s$, which represents a month.

For the phase and gain margin, the next Bode diagram was availed in the metrics of the requisites:

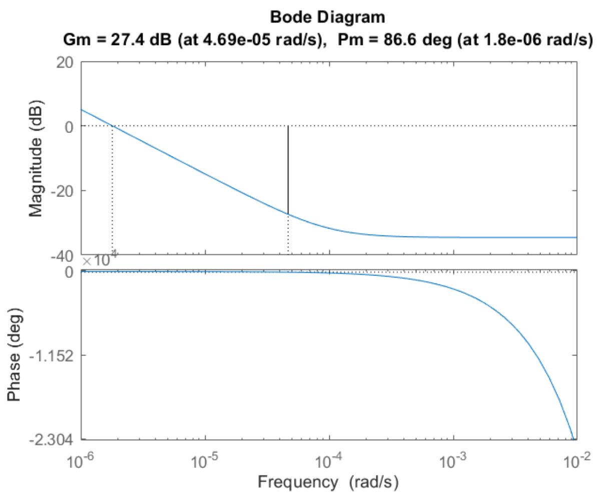

Figure 9 – Bode diagram for the open loop control optimal system

Note that neither the phase margin achieved the goal 120 degrees, or the gain margin achieved the goal 30 dB, but each of margins obtained by the optimization was 27.4 dB for the gain margin and 86.6 degrees for the phase margin, which both was good margins for the industry, although not achieving the initial margins.

The bandwidth was 1.9103e-06 Hz, which prove to achieve the goal $\omega_b <= 2.4241 \cdot 10^{-6} Hz$. So, its concluded that the controller founded was good for the implementation, as gives good margins for error and satisfy the others requisites. So, once the controller was projected, the simulation with the discrete controller was done.

B. Simulation

The initial noise has a mean of 0 and a standard deviation of 0.05 of the maximum occupancy. The parameter a was obtained using data from the 600 block of Beach Street, Fisherman's Wharf.

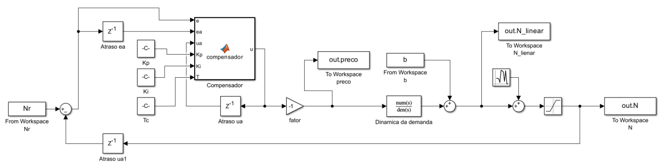

Figure 10 – Simulink block diagram.

The noise represents small variations in demand over short periods throughout the days, caused by factors that cannot be modeled. The block diagram in Simulink provides a visual representation of the system used for simulation.

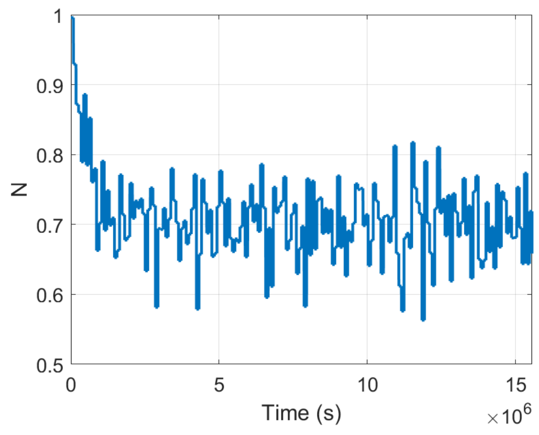

Figure 11 – Actual percentage of occupied parking spaces (occupancy) over time.

The controller was able to, without any additional filtering, be resilient to noise due to the integrator. We achieved the expected occupancy (0.7) without any apparent overshoot, meeting our requirements. The rise time is around one month, but we do not consider this delay negative; on the contrary, we required the system to be slow to allow people to adjust.

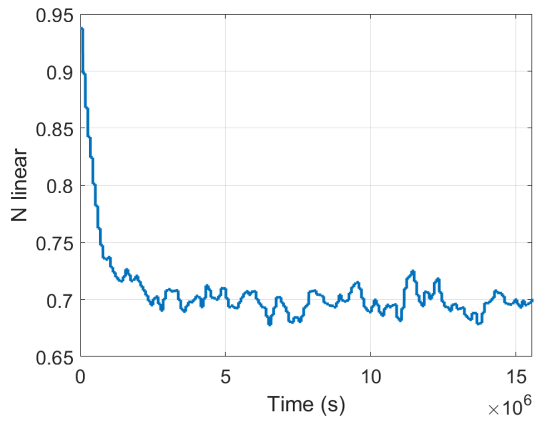

Figure 12 – Percentage of occupied parking spaces (occupancy) over time before passing through the saturation and noise blocks in Simulink.

This figure shows the occupancy percentage before the effects of saturation and noise are applied. The integrator's influence can be observed here, where the control action smooths out the variations due to noise.

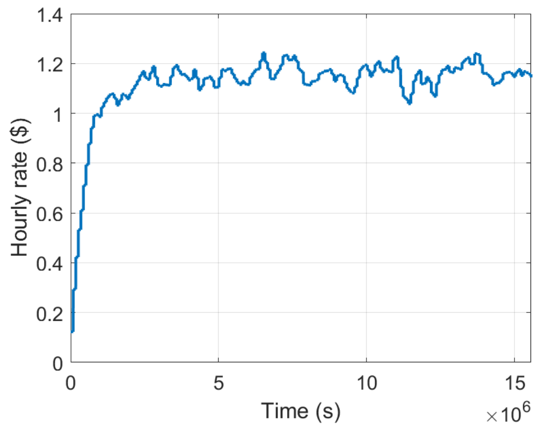

Figure 13 – Hourly parking price (in dollars) over time. The price converged to \$1.15 in about 30 days.

The hourly parking price adjusts over time to maintain the desired occupancy level. This evaluation was performed with a controller designed based on the same plant used for the simulation. To robustly assess the design without data leakage, it is necessary to use a plant with different data to evaluate if it will be effective as well.

1) Changes in the noise: The noise standard deviation was increased to 0.2 of the occupancy.

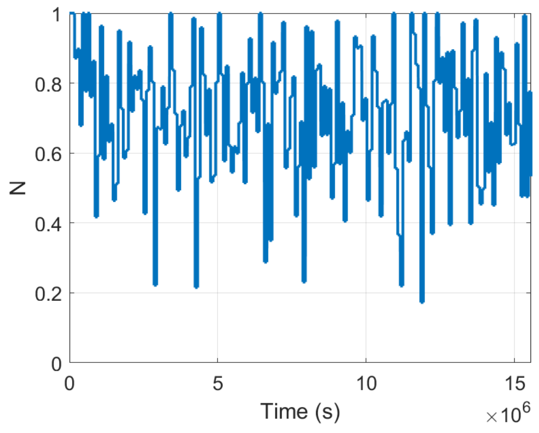

Figure 14 – Actual percentage of occupied parking spaces (occupancy) over time of simulation with increased noise.

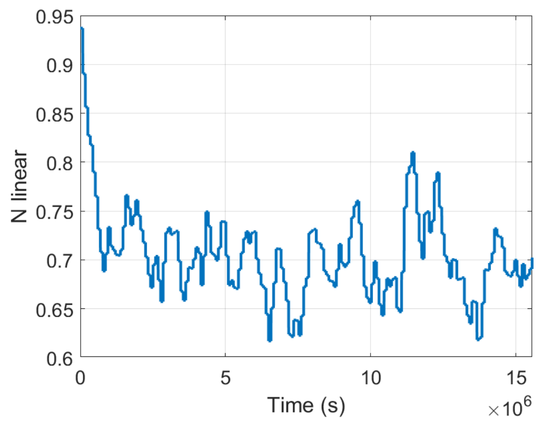

Figure 15 – Percentage of occupied parking spaces (occupancy) over time before passing through the saturation and noise blocks in Simulink of simulation with increased noise.

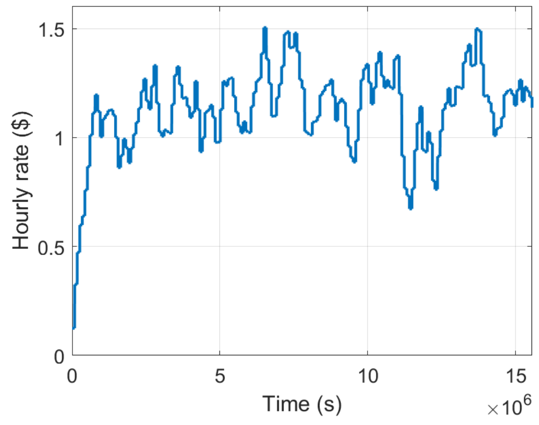

Figure 16 – Hourly parking price (in dollars) over time of simulation with increased noise.

2) System loop variation: The standard deviation of the noise was reduced back to the original value (0.05), and the demand curve was replaced with an approximation using data from the 200 block of Drumm Street, downtown. The controller was trained using the previous dataset from the 600 block of Beach Street, Fisherman's Wharf.

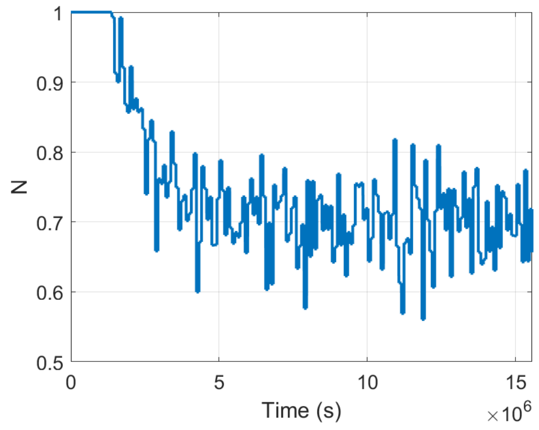

Figure 17 – Actual percentage of occupied parking spaces (occupancy) over time using new data.

The actual percentage of occupied parking spaces shows that the controller can handle the new demand data, but with different characteristics compared to the previous dataset.

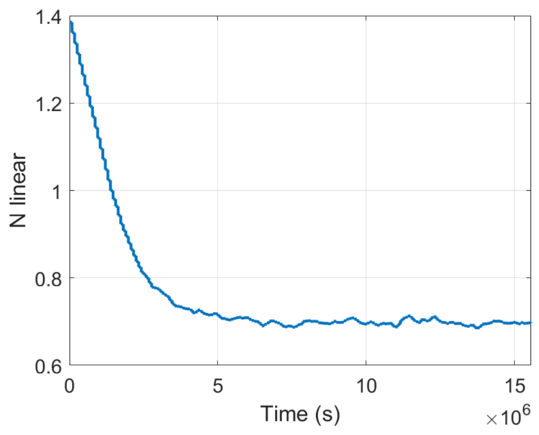

Figure 18 – Percentage of occupied parking spaces (occupancy) over time before passing through the saturation and noise blocks in Simulink using new data.

Before applying saturation and noise, the occupancy curve shows a slower convergence compared to the previous data. This slower convergence, taking about three months, is due to the controller being designed based on the previous dataset. The pricing strategy fails to account for the higher demand, resulting in a less aggressive price increase. Additionally, natural saturation of occupancy (cannot exceed 100%) limits the "rise speed," but this is primarily due to the initial training data.

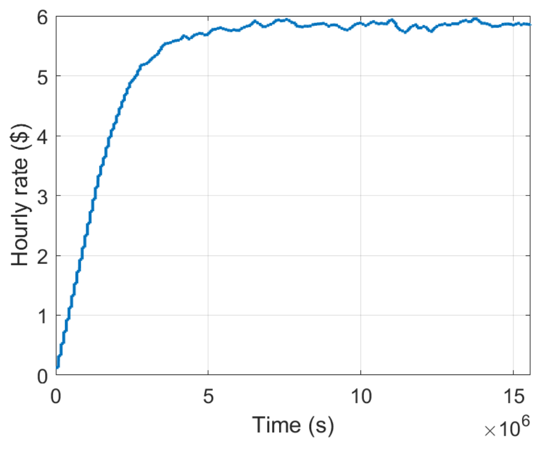

Figure 19 – Hourly parking price (in dollars) over time using new data.

The hourly parking price, now converging to \$5.9, reflects the higher demand. If the situation were reversed (designing with higher price data and simulating with lower price data), the rise would be more violent and converge in less time. This shows that the rise time is highly dependent on the plant data used for the design, making it essential to use the correct dataset. Regardless of the rise time, the steady-state error remains zero.

In conclusion, while the rise time varies significantly depending on the training data, the system still meets the requirement of zero steady-state error, demonstrating the robustness of the controller in different scenarios, and this is guaranteed by our phase margin and gain margin requirements.

V. CONCLUSION

Based on the comprehensive study conducted on dynamic parking pricing using a PI controller with anti-windup, several key conclusions can be drawn. The optimized controller, derived through advanced optimization techniques like CMA-ES and Nelder-Mead, effectively regulated parking prices to achieve a target occupancy rate of 70%. Despite not fully meeting the initial phase and gain margin targets of 120 degrees and 30 dB respectively, the achieved margins of 86.6 degrees and 27.4 dB proved robust for practical implementation, ensuring stable and effective control of parking occupancy.

Simulation results demonstrated the controller's resilience to varying levels of noise in demand data, showcasing its ability to maintain desired parking occupancy and adjust prices accordingly over extended periods. This robustness was particularly evident when tested with different datasets, highlighting the importance of appropriately training the controller with representative data to optimize convergence time and pricing strategy effectiveness.

In conclusion, the developed control system offers a promising solution for dynamically managing parking prices to alleviate congestion and optimize space utilization. Future refinements could focus on further enhancing the controller's performance under diverse environmental conditions and integrating real-time data feedback mechanisms to continuously improve operational efficiency and user satisfaction.

REFERENCES

[1] City Beautiful. *How Much Should a Parking Space Cost?* YouTube. Published January 20, 2024. https://www.youtube.com/watch?v=78ffidI7__E

[2] Pierce G, Shoup D. *Getting the prices right*. Journal of the American Planning Association. 2013;79(1):67-81. doi:10.1080/01944363.2013.787307

[3] Wheeler K. *Citywide meter rate adjustments*. SFMTA. Published July 15, 2024. https://www.sfmta.com/notices/citywide-meter-rate-adjustments

[4] K. J. Åström, T. Hägglund, *PID Controllers: Theory, Design, and Tuning*, Instrument Society of America, 1995.

[5] G. F. Franklin, J. D. Powell, A. Emami-Naeini, *Feedback Control of Dynamic Systems*, Addison-Wesley, 1994.

[6] R. C. Dorf, R. H. Bishop, *Modern Control Systems*, Addison-Wesley, 2001.

[7] N. S. Nise, *Control Systems Engineering*, Wiley, 2004.

[8] P. Albertos, I. Mareels, *Feedback and Control for Everyone*, Springer, 2010.

[9] J. A. Nelder, R. Mead, *A Simplex Method for Function Minimization*, Computer Journal, 1965.

[10] N. Hansen, A. Ostermeier, *Completely Derandomized Self-Adaptation in Evolution Strategies*, Evolutionary Computation, 2001.